PATENT APPLICATION ENTITLED

"SYSTEM AND METHOD OF DISCOVERING INFORMATION"

By Inventor:

Jeffrey S. Larson

and

Gary Cole

Date Filed:

December 6, 2001

Attorney for Applicant:
Customer ID: 25094
GRAY CARY WARE & FREIDENRICH LLP
1221 South MoPac Expressway, Suite 400
Austin, Texas 78746
Attn: John L. Adair
(512) 457-7142 (Phone)
(512) 457-7001 (Fax)

SYSTEM AND METHOD OF DISCOVERING INFORMATION

RELATED INFORMATION

[0001] This application claims priority under 35 U.S.C. §119(e) to United States Provisional Patent Application serial No. 60/251,627 entitled "System and Method for Automatically Discovering Information," filed on December 6, 2000, which is hereby fully incorporated by reference. This application also claims priority under 35 U.S.C. §119(e) to United States Provisional Patent Application serial No. 60/251,952 entitled "System and Method for Tracking Information in a Pointer-Driven Repository," filed on December 7, 2000, which is hereby fully incorporated by reference.

TECHNICAL FIELD OF THE INVENTION

[0002] The present invention relates generally to systems and methods of systems management. More particularly, the present invention relates to a system and method for the discovery of data. Even more particularly, the present invention relates to a system and method for the discovery of user accounts and other information objects on a network.

BACKGROUND OF THE INVENTION

[0003] Processing and storage of electronic data is now essential to the daily operation of most organizations. With the advent of networking technology, organizations that utilize electronic data processing are becoming increasingly reliant upon "enterprise" computer networks in which processing and storage are distributed over a number of heterogeneous interconnected computers. In many enterprise systems, a member of the organization will have access to multiple resources across the system. For example, an employee of a corporation may a use an email account, a Windows NT account, and a Unix account to access and process data stored on the enterprise system. Additionally, organizations will often wish to provide external users, such as distributors, business partners and suppliers, with accounts granting limited access to the data stored on the enterprise system. The administrative overhead required to manage the internal and external accounts often becomes more difficult to manage than the data that is actually of interest to the organization. This can lead to decreases in system efficiency and to high support costs.

[0004] Consequently, organizations are becoming increasingly interested in efficient systems management as it can provide, among other benefits, reduced information technology ("IT") costs and increased efficiency in setting up and managing enterprise data. Currently, however, providing efficient systems management for enterprise computer networks, particularly those that contain legacy data, is a quixotic task. This is partly because many organizations, over time, have developed networks including a variety of heterogeneous computer systems storing myriad different data types. Further adding to the complexity of managing enterprise networks, organizations often store inconsistent data across the network. As just one example of data inconsistencies, a company may store one home phone number for an employee at a corporate human resources ("HR") mainframe while storing a different home phone number at a departmental mainframe. Because the two mainframes may be heterogeneous (e.g., employ different hardware, operating systems, protocols, tools and/or applications), synchronizing between the two resources to eliminate inconsistencies can prove difficult.

[0005] Most prior art systems management techniques address these difficulties by centralizing data. Profile-based management systems, directory-based management systems, and meta-directories offer various approaches to centralizing data storage. FIGURE 1 illustrates the limitations of prior art systems that rely on centralization of data. FIGURE 1 is a

diagrammatic representation of computer system 100 comprising an administrative system 110, including a centralized database 112, and resources including an email server 120 (such as a Microsoft Exchange server), a Unix system 125, a Windows NT system 130 and a mainframe 135. The resources are interconnected to each other and are connected to administrative system 110 via a network 145. Each resource can contain a collection of data items that represent entities or individuals. For example, e-mail server 120 can contain a collection of email accounts 150, Unix system 125 can contain a collection of Unix accounts 155, Widows NT system 130 can contain a collection of Windows NT accounts 160 and mainframe 135 can contain a collection of data records 165.

[0006] These collections of data represent each resource's "view" of an individual or entity. In the case of an employee Jane Doe, for example, email server 120 may refer to her as janed (i.e., her e-mail user name), Unix system 125 may refer to her as JaneD (i.e., her Unix account user name) or by her Unix identification ("UID"), and Windows NT system 130 may refer to Jane Doe as JANED (i.e., her windows NT account user name). In addition to account information allowing Jane Doe to access data on computer system 100, information such as Jane Doe's department code, time keeper number, salary rate, and employee identification can be stored at mainframe 135. This may be information that is not personally used by Jane Doe, but it is used, instead, by her managers or other personnel. Thus, mainframe 135 would also maintain an identity for Jane Doe, based on her employee record, which could, for example, be stored under JANE_D.

[0007] To illustrate the shortcomings of prior art systems that rely on centralization of data, assume that employee Jane Doe marries and changes her last name to Smith. One method of updating Jane Doe's name on system 100 would be to separately enter the updated information at each system. For an organization having a large number of users and/or a highly distributed computer system 100, this can be impractical. To ameliorate the inefficiencies of separately entering information at each resource, one prior art system replicates all the information identifying individuals or entities in a centralized database 112 (represented by replicated data 175). Thus the collection of email accounts 150, the collection of Unix accounts 155, the collection of Windows NT accounts 160 and the collection of data records 165 are typically replicated at centralized database 112. When a change is made to the data, the change can be entered to replicated data 175 and can then be pushed out to each of the

resources. In the case of Jane Doe, then, replicated data 175 is modified to account for her name change, and the replicated data can then be pushed out to one or more resources. In the case of an individual such as Jane Doe, the replicated data, thus, contains a "master copy" of her data.

[0008] While a system having a centralized database helps ensure data consistency for data entered through administrative system 110 and pushed out to each resource, it has several shortcomings. One such limitation involves the resolution of inconsistencies between data changed at the individual resources. Continuing with the example of newlywed Jane Doe; if Jane Doe changes her last name to Smith, her name may be inadvertently changed to Smyth at mainframe 135, while her name is changed to Smith at email server 120. When data from the resources is copied to centralized database 112, there will be three names for the same employee on computer system 100: Jane Doe, Jane Smyth and Jane Smith. Administrative system 110 must determine if a name change is actually appropriate and which of the changes is appropriate. Once the specific change is selected, the change is distributed to the resources, overwriting local changes (or lack of changes) made at each resource. Thus, for example, if Jane Smyth was arbitrarily selected as the correct change, Jane Smyth would be distributed to each of the resources, overwriting the correct name, Jane Smith.

[0009] Furthermore, if different resources are controlled by different groups within the organization, the decision to favor one resource over another can lead to political tension within the organization. As an additional limitation of this prior art system, in a large enough computer system 100, some subset of the resources will be unavailable at any given time due to connectivity issues or other technical problems. Therefore, only some of the resources will be updated, causing additional inconsistencies in Jane Doe's data.

[0010] In order to implement a centralized database approach, prior art systems typically rely on an overall systems administrator or manager to identify resources used by members of the organization. This administrator's view of which resources members of the organization require, however, is typically limited and the administrator generally has little idea of which resources are used on a department-by-department basis or of the resources used on an individual level. Furthermore, prior art systems typically do not automate the process of retrieving, mapping, correlating and merging information from multiple heterogeneous sources. Because of this, administrators must typically write ad hoc queries and scripts to extract, map,

correlate, merge and upload information into the centralized database 112, making implementation of the centralized database tedious.

[0011] Centralization of data typically requires replicating at least some subset of the data being managed. This type of system scales poorly because of the large amount of data that must be stored at the centralized database 112 and it further introduces problems with synchronizing the centralized database 112 with the resources. Furthermore, because these systems require data to be copied repeatedly back and forth from the resources to the centralized database 112, significant bandwidth demands are inflicted upon the network. As yet another shortcoming, manually locating and organization data from a number of resources typically requires significant investments of time and money. Thus, prior art systems are generally expensive and inefficient.

SUMMARY OF THE INVENTION

[0012] Embodiments of the present invention comprise a system and method for discovering data on a network that substantially reduces or eliminates problems associated with previous data discovery systems and methods. More particularly, embodiments of the present invention provide a system and method for automatically discovering information on a network.

[0013] Embodiments of the present invention can comprise the steps of (i) defining at least one source resource, such as an email server, containing a set of information objects (e.g., accounts), wherein the set of information objects defines a set of users; (ii) extracting said set of users from said source resource; (iii) defining an additional resource containing a second set of information objects, wherein each information object from said second set of information objects corresponds to a user from said set of users; (iv) automatically discovering said second set of information objects from said additional resource; and (v) associating each information object from said second set of information objects with the corresponding user from said set of users.

[0014] Another embodiment of the present invention comprises (i) defining a first resource containing information objects defining a set of users; (ii) discovering said information objects based on said first resource definition; (iii) associating each of said information objects with a user from said set of users and with said first resource.

[0015] Yet another embodiment of the method of the present invention includes a method comprises the steps of (i) receiving a first resource definition from a first administrator, wherein said first resource contains a first set of information objects defining at least one user from a set of users; (ii) receiving a second resource definition from a second administrator, wherein said second resource contains a second set of information objects defining at least one user from said set of users; (iii) discovering said first set of information objects from said first resource;(iv) associating each information object from said first set of information objects with at least one user from said set of users and with said first resource; (v) discovering said second set of information objects from said second resource; and (vi) associating each information object from said second set of information objects with at least one user from said set of users and with said second resource.

[0016] Yet another embodiment of the present invention comprises the steps of (i) defining a plurality of resources, wherein each of the plurality of resources contains a set of information objects defining at least one user from a set of users; (ii) discovering said set of

information objects from each of the plurality of resources; and (iii) associating each information object from said set of information objects with a user from said set of users and with the resource from which the corresponding information object was discovered.

[0017] Embodiments of the present invention provide an advantage over prior art systems and methods of data management by locating data on a computer network in an automated manner, thereby being more efficient and cost effective.

[0018] Additionally, embodiments of the present invention provide an advantage over prior art systems and methods of data management by utilizing a phased approach to data discovery. A phased approach to data discovery allows for more accurate and economical discovery of data than prior art systems.

[0019] Additionally, embodiments of the present invention provide an advantage over prior art systems and methods of data management by supporting delegation of discovery. Delegating data discovery allows for more accurate and economical discovery of data than prior art systems.

[0020] As an additional advantage provided by embodiments of the present invention, the present invention can reduce bandwidth requirements for managing user accounts.

[0021] Additionally, embodiments of the present invention provide a significant advantage over the prior art by being highly scalable.

BRIEF DESCRIPTION OF THE DRAWINGS

[0022] A more complete understanding of the present invention and the advantages thereof may be acquired by referring to the following description, taken in conjunction with the accompanying drawings in which like reference numbers indicate like features and wherein:

[0023] FIGURE 1 illustrates a prior art system for systems management in which data is replicated at a centralized database;

[0024] FIGURE 2 illustrates one embodiment of a computer system in which the teachings of the present invention can be implemented;

[0025] FIGURE 3 illustrates a system of account management according to one embodiment of the present invention;

[0026] FIGURE 4 illustrates a schema map according to one embodiment of the present invention; and

[0027] FIGURE 5 is a flow chart illustrating one embodiment of a method for discovering users and information objects according to the present invention.

DETAILED DESCRIPTION OF THE INVENTION

[0028] Preferred embodiments of the present invention are illustrated in the FIGUREs, like numerals being used to refer to like and corresponding parts.

[0029] Embodiments of the present invention comprise a system and method for the automated discovery of data items on a network. Embodiments of the present invention can comprise delegated discovery, wherein the steps of defining resources for the discovery process are delegated among a plurality of administrators and end users. Delegation of discovery allows the discovery process to be configured by those most knowledgeable about resource usage and thereby increases the accuracy of discovery. Furthermore, embodiments of the present invention automate the discovery process, thereby decreasing implementation time and complexity.

[0030] For the purposes of this application, the term "resource" can mean a system or application accessible by a network that defines information objects related to its management or operation. For example a Unix system can be a resource that defines user accounts (i.e., information objects), typically in its /etc/passwd file, a Windows system can be a resource that defines user accounts in its User Manager application and a DBMS system can be a resource that defines user accounts in special table or tables. It should be noted that a resource can comprise a particular computer system (distributed or undistributed), an application on a computer system, or an application distributed across several computer systems. A "user" can be a human, a programmatic or a computer system that uses the resources. An "information object" can be a collection of one or more pieces of data that can represent a single entity or identity. In other words, an information object (such as an account) can represent a resource's "view" of an entity (such as a user). An attribute can be a single piece of information (e.g., a data structure with a name, a data type and zero, one or more values) that constitutes at least part of an information object. A "resource attribute" can be an attribute residing on a resource. A "schema" can be the structure of and relationships between classes of information objects on a resource. The schema can include the set of attribute definitions the resource uses to describe each information object. A "virtual identity" can be a composite identity of a user based on one or more information objects discovered from one or more resources. A "virtual information object" can be a collection of attributes constructed based on the information objects associated with a specific virtual identity.

[0031] FIGURE 2 illustrates a system 200 in which one embodiment of the present invention can be implemented. System 200 can comprise a plurality of resources (here, indicated as resource 210, resource 212, resource 214, resource 215, resource 216 and resource 218). Each resource can include a computer system (either discrete or distributed) and/or an application on a computer system. While system 200 includes multiple resources, it should be understood that the present invention can be implemented in a system having only one resource. System 200 can also include an administrative system 220 operable to access the resources via network 225, which can be a LAN, WAN, global area network (e.g., the internet, wireless network) or any other electronic communications network known in the art. Administrative system 220 can comprise a computer processor 230, a computer readable memory 235 (e.g., RAM, ROM, computer readable magnetic storage device and/or other computer readable memories known in the art) and a software program 240 executable by computer processor 230 to implement automated discovery of information objects and users from the resources. While administrative system 220 is shown as a discrete system, it should be understood that administrative system 220 can be distributed and/or implemented in the same physical unit(s) as one or more of the resources.

[0032] Each resource can define information objects related to its management or operation. By way of example, resource 210 can comprise an Oracle database system including employee records 242 as information objects; resource 212 can comprise first Unix system containing a first set of Unix user accounts 244; resource 214 can comprise a Windows NT system containing a set of NT user accounts 246; resource 215 can comprise an email server (e.g., Microsoft Exchange Server) containing email accounts 248; resource 216 can comprise a second Unix system containing a second set of Unix user accounts 250; and resource 218 can comprise a third Unix system containing a third set of Unix user accounts 252. The resource accounts (i.e. information objects) represent each resource's "view" of a particular user. In other words, each resource account (and the attributes that make up that account) represent a user within the scope of the resource. Thus, for example, to email server 215, user Jane Doe is:

User name: Jane Doe
account name: jdoe
password: *********
employee id: 12345

[0033] It should be noted that not all users will have access to all the resources of system 200. Further, different administrators in an organization can have different levels of knowledge about whether particular users or types of users, as will be discussed in greater detail below, utilize a particular resource. For example, administrator 260 may know that all members of the organization have records on Oracle database 210, Unix accounts on first Unix system 212, user accounts on Windows NT system 214 and email accounts on email server 215. In addition, first departmental administrator 262 may have knowledge about which users have accounts on second Unix system 216 and second departmental administrator 264 may have knowledge about which users have accounts on third Unix system 218.

[0034] As will be discussed in greater detail below, based on information provided by one or more of the administrators and/or by the end-user, program 240 can discover and correlate accounts on system 200 and create an identity index associating these accounts with users. Thus, embodiments of the present invention can "collect" resource-specific identities (e.g., accounts) to construct a virtual information object.

[0035] Thus, with varying degrees of input from the administrators (e.g., administrator 260, first departmental administrator 262 and second departmental administrator 264), software program 240 can locate information objects and data items on the resources, correlate the information objects to users and create an index associating each user with information objects and resources. Furthermore, software program 240 can retrieve attributes from each resource and present the data items in a canonical format without replicating the attributes at a centralized database (i.e., without storing the values for the attributes in nonvolatile memory). While the present invention will be described primarily in terms of managing computer accounts of various formats accessed by human users, the teachings of the present invention are equally applicable to discovering any information objects and users. Thus, for example, embodiments of the present invention are configurable to discover routing tables that reference a particular network element and so on.

[0036] FIGURE 3 is a diagrammatic representation of a management system 300 for managing user accounts according to one embodiment of the present invention. Management system 300 can comprise software program 240 for discovering users and resource accounts from resources, an identity index 310 to associate users with resource accounts, schema maps (e.g., schema map 330, schema map 332 and schema map 334, discussed below) to associate resource attributes with virtual attributes, and a composite view 319 to present a "view" of a user according to one or more resources. Software program 240, as noted earlier, identity index 310 and the schema maps can be stored in a memory 235 of administrative system 220 (see, FIGURE 2).

[0037] Software program 240 can invoke resource adapter modules 325 to communicate with Oracle database system 210, first Unix system 212, NT system 214, email server 215, second Unix system 216 and third Unix system 218. One resource adapter module 325 is typically operable to communicate with a particular resource type (e.g., one resource adapter 325 can be invoked to communicate with all the Unix systems), however separate resource adapter modules 325 could also be used for each resource, as illustrated in FIGURE 3. As would be understood by one of ordinary skill in the art, resource adapter modules can also be configured to communicate with any resource type. From the resources, software program 240, as will be discussed in conjunction with FIGURE 5, can extract a list of resource users from one or more of the resources, create virtual identities for the users and associate resource accounts with users in identity index 310.

[0038] In the embodiment illustrated in FIGURE 3, identity index 310 can contain a number of virtual identities (e.g., virtual identities 312, 314 and 315). The virtual identities 312, 314 and 315 can be created by software program 240 based on users discovered on the resources by software program 240, or they can be independently created (e.g., by an administrator). Each virtual identity 312, 314 and 315 can contain a virtual identity name 345 and a list of resource accounts 350 associated with the user corresponding to the virtual identity. For example, if employee Jane Doe has an account on Oracle database system 210, first Unix system 212 and NT system 214, the native key for those accounts (e.g., the account ID) and the resource on which those accounts are located can be stored in Jane Doe's virtual identity. Additional resource attributes can be stored in a virtual identity depending on the configuration of system 300 (e.g., if the organization wishes the capability to search identity

index 310 by user name, they may want to store attributes such as "firstname," "lastname," or "fullname").

[0039] Additionally, identity index 310 can include a set of resource definitions, such as resource definitions 320, 322, and 324. In FIGURE 3, resource definition 320 corresponds to Oracle database system 210, resource definition 322 corresponds to first Unix system 212 and resource definition 324 corresponds to NT system 214. Each resource definition can contain the information needed by software program 240 to connect to the corresponding resource. One embodiment of an identity index 310 of the system and methods of the present invention is described in greater detail in United States Patent Application serial No. _____, entitled "A System and Method for Managing Information Objects" filed December 6, 2001, which is hereby fully incorporated by reference.

[0040] Additionally, each resource definition can include a schema map. Thus resource definition 320 can include schema map 330, resource definition 322 can include schema map 332, and resource definition 324 can include schema map 334. A schema map associates resource-specific information object (e.g., account) attributes to a virtual attribute defined in the schema map. In this manner, the schema map allows software program 240 to display attributes values (i.e. data items) from different resources in a common format. For example, attributes from different resources can be mapped to the same virtual attribute. Thus for example, if Oracle database system 210 contained an attribute named "phone" and first Unix system 212 contained an attribute name "ph_num," schema maps 330 and 332 could, respectively, map the attributes to a virtual attribute named "Phone Number."

[0041] In one embodiment of the present invention, the values for the virtual attribute "Phone Number" are not be stored in identity index 310. Instead, when a user or administrator accesses the user's virtual identity, software program 240 can access the attributes "ph_num" and "phone" (from resources 212 and 210, respectively), store the values in RAM (e.g., construct a virtual information object) and display the values under "Phone Number" in composite view 319. If the values match, composite view 319 can display only one value (in a graphical user interface, for example), or if the values do not match, the user or administrator can be given the option to select the correct value. Furthermore, if the administrator or user makes a change to a virtual attribute associated with a user, the change can be pushed out to the appropriate resource-specific attributes based on the schema maps. The schema maps,

thus, provide a way to represent resource-specific attributes in a standard or canonical format (i.e. to construct a virtual information object) and to map changes back to resource specifications. FIGURE 4 illustrates one embodiment of a schema map.

[0042] Composite view 319 can comprise a view of a user's accounts as defined by a schema map. Based on the schema map(s), composite view 319 can contain virtual attribute values derived from the resource-specific attributes for a user and can be presented to administrators and users through a graphical user interface. The organization implementing system 300 can determine various levels of authority for administrators and users to view and or modify certain virtual attributes. Thus, the graphical user interface may display different information to a user or administrator depending on his or her status. When a user or administrator modifies the value for a virtual attribute, software program 240 can push the change to the virtual attribute to the resource-specific attributes associated with the modified attribute in the schema map. Thus, data across resources can be synchronized without replicating the attributes at a central database.

[0043] FIGURE 4, illustrates one embodiment of a schema map (e.g., schema map 330 of FIGURE 3). A schema map for a particular resource, such as schema map 330, can contain a virtual attribute 410, resource attribute 420 and an attribute type 430. In the example of FIGURE 4, one virtual attribute 410 is "Phone Number," whereas the corresponding resource attribute 420 is "phone." When software program 240 reads the resource specific attribute "phone" associated with a user's resource account, software program 240 can map the value for "phone" to the virtual attribute "Phone Number" for the user's virtual identity. It should be noted that the value for "Phone Number" might not be saved in nonvolatile memory (e.g., with identity index 310), but may instead only be saved in RAM while the virtual identity is being manipulated or created.

[0044] In schema map 330, attribute type 430 can comprise "string" and "Boolean" or other possible attribute types as would be understood by those of ordinary skill in the art. The attribute type 430 field can be used to map different types of attributes together. It should be noted that in some embodiments of the present invention, not every resource attribute will be mapped to a virtual attribute. These unmapped attributes will typically not appear in composite view 319, identity index 310 or schema map 330. This might occur, for example, if an

organization implementing an embodiment of this invention does not want particular attributes to be discoverable, such as employees' social security numbers.

[0045] It should be further noted that if all instances of a resource type are the same, a schema map can be defined for a resource type rather than on a resource by resource basis. As such, fore example, there would be one schema map for NT systems, on schema map for Unix, one schema map for LDAP and so on. However, in many cases an organization may configure the same resource in several ways. For example, an organization may store different types of data in an NT systems "description" field on different systems. Thus, each resource will typically require a unique schema map.

[0046] FIGURE 5 is a flow chart illustrating one embodiment of the method of this invention for discovering information objects (e.g., accounts) and users on system 200. The process of discovery can occur in phases, beginning with "central" definition and discovery, followed by delegated definition and discovery. While these phases will be discussed separately, they can, in practice, overlap. Furthermore, the reader should understand that the phases, while presented as a sequence of tasks, are flexible and each phase and/or step can be practiced to a greater or lesser extent depending on the implementation of embodiments of the present invention. It should be further understood that the steps of FIGURE 5 can involve both human action by an administrator and automated software processing performed by software program 240.

[0047] In steps 500-515 an administrator can identify and define resources. During the first phase, this will typically be done by the administrator or by a set of administrators who have the most knowledge of resource usage on system 200 (e.g., a central administrator such as administrator 260). Thus, for example, as administrator 260 has knowledge of the resources most commonly used, administrator 260 can identify those resources (e.g., he can identify Oracle database system 210, first Unix system 212, NT system 214 and email server 215 as resources). Additionally, to "define" a resource, administrator 260, at step 510, can provide a set of identifying information, such as a name for each resource, the resource type, hostname, port number, a resource username, resource password and other such information to software program 240 through, for example, a graphical user interface. To illustrate, for Unix system 212 administrator 260 might provide the following information:

Resource name= “reso01”
Resource type = “solaris”
Hostname = “resource1.organization.com”
Port =”23”
Protocol= “telnet”
Username = “root”
Password =“********”

[0048] As a further example, administrator 260 could provide the following information for Windows NT system 214:

Resource name=”reso02”
Resource type = “nt”
Hostname = resource2.organization.com
Port =”789”
Domain= “Topeka”
Username = “administrator”
Password =“********”

[0049] As a final example, for Oracle database system 210, administrator 260 could provide the following:

Resource name=”reso03”
Resource type = “Oracle”
jdbcDriver = ”com.Oracle.jdbc.DriverManager”
url = ”jdbc:Oracle//resource3.organization.com:1789/db03
Username = ”System”
Password = ”*******”

As would be understood by one of ordinary skill in the art, administrator 260 can provide information sufficient for software program 240 to connect to and communicate with a resource. As would be further understood, depending on the type of resource employed, different types of information may be required to define a particular resource. In one embodiment of the present invention, the resource definitions provided in step 510 can be stored in identity index 310.

[0050] At step 512 administrator 260 can define a "schema map" which associates different resource-specific information objects (e.g., accounts) to a virtual attribute defined in the schema map. The schema map allows software program 240 to display and manipulate attribute values from different resources in a common format. It should be noted that administrator 260 generally knows the schema of each resource (e.g., the structures of information objects on the database) or knows how to derive the schema for each resource. Based on the schema for each resource, the administrator can map attributes from different resources to the same virtual attribute at administrative system 220 (e.g. can define schema maps). In one embodiment of the present invention, not every attribute of an account is mapped to a virtual attribute in the schema map. Instead, only those attributes that the organization wishes to centrally manage are so mapped. As one example of the functionality of a schema map, administrator 260 can map an attribute in Oracle system 210 named "phone" and an attribute on first Unix system 214 named "ph_num" to a virtual attribute on administrative system 220 named "Phone Number."

[0051] Again, it should be noted that if all instances of a resource type are the same, a schema map can be defined for a resource type rather than on a resource by resource basis. For example, there can be one schema map for NT systems, on schema map for Unix, one schema map for LDAP and so on. However, in many cases an organization may use the same resource type differently. For example, different departments within an organization may record different types of information in the "description" field of NT user accounts. Thus, each resource of the same resource type may be unique and administrator 260 can define a separate resource map for each resource. An embodiment of a schema map is illustrated in FIGURE 4.

[0052] As a further step in defining a resource, administrator 260, at step 515, can define an identity template used to generate information object names on that resource. The identity template can be used to generate a resource-specific names (e.g., an account name) given a virtual identity and its attributes. For example, if a virtual identity for a user already exists in identity index 310, the identity template can be used to generate a possible account name for that user on a resource, and then check the resource to see whether an account with that account name exists. The identity template can automatically associate an account discovered under the generate account name with the user. This "directed discovery" can be a very efficient way to associate additional user accounts with a virtual identity.

[0053] At step 516, administrator 260 can define resource name rules for each resource including account name rules and user name rules. In an account name rule, given the name of a user, software program 240 can generate a list of possible account names on the resource for a user. Account name rules can typically generate multiple account names, whereas the identity template defined at step 515 typically generates only one account name for a user. Furthermore, accounts discovered based on account name rules may require additional correlation, as will be described below. With a user name rule, given a particular resource account name, software program 240 can generate a list of possible users. For example, given the name "Jane Doe," and account name rule for email server 215 could generate the following possible matches for account names:

jdoe	$(substr(1,1,$first))$(substr(1,4,$last))
doej	$(substr(1,4,$last))$(substr(1,1,$first))
jdoe2	$(substr(1,1,$first))$(substr(1,4,$last))[1..9]

[0054] Administrator 260, at step 518, can also define a set of resource correlation rules. With an account correlation rule, given a set of resource accounts, software program 240 can test whether any of the accounts match a particular user based on the resource account attributes and a correlation key specified by administrator 260. For example, a correlation rule may be based on whether a particular resource has an attribute an employee number matching a known employee number for a user (e.g., as stored in identity index 310 or on another resource already associated with the user). The correlation rule can also include a formula involving manipulation of resource account attributes. With a user correlation rule, given a set of users, software program 240 can determine whether any of the users discovered match the account.

[0055] At step 520, administrator 260 can define one or more source resources. A source resource refers to a resource from which one or more users of system 200 may be discovered. If a complete list of users can be discovered from a single resource (e.g., a central directory) administrator 260 can configure software program 240 to read information objects (e.g., accounts) from that resource first. If a list of users can be discovered from non-overlapping resources, then administrator 260 can configure software program 240 to read the user names from the non-overlapping resources. If a complete list of users can be discovered from resources that overlap but for which there is a correlation key (e.g., at least one data item

associated with each user name that is common to both resources), administrator 260 can configure software program 240 to read the user names from the overlapping resources and detect duplicate names based on the correlation key.

[0056] For example, if one email server contained email accounts for all the programmers in an organization, while another email server contained email accounts for all the managers in the organization, but some of the managers were also programmers, the two sets of accounts would overlap. Administrator 260 could define a correlation rule to correlate the two sets of account and reject duplicates based on attributes such as employee name, social security number, employee number and so on. If, on the other hand, a list of users can be discovered from resources that overlap and for which there is no correlation key, administrator 260 can determine how a unique set of user can be discovered from the resources most easily. If a complete list of users can not be discovered, then administrator 260 can configure software program 240 to discover only a partial list of users.

[0057] From the source resource(s) defined by the administrator 260, software program 240, at step 522, can discover user identities. In other words, software program 240 can isolate the unique set of users represented by the accounts at the source resource(s). If a virtual identity does not exist in identity index 310 for a particular user discovered from the source resource, software program 240 can create a virtual identity based on one or more arbitrary attributes of the user's account found on the source resource (e.g., the user name) and according to the schema map for the source resource.

[0058] For example if the user Jane Doe is found on the source resource, a virtual identity can be created for Jane Doe by, for example, mapping the user name Jane Doe to a new virtual user name (e.g., equal to the account name, or as generated by any user name rule defined for the source resource). During importation of accounts, software program 240 can extract the account name. Additionally, software program 240 can also extract the account attribute values and convert them into a standard format according to the resource schema map for the source resource (i.e., can translate the account attributes into virtual attributes). In one embodiment of the present invention, some of the attributes are stored under a user's virtual identity in identity index 310 while the rest are kept only in volatile memory until software program 240 is finished manipulating them. For example, the attributes stored under a user's virtual identity could comprise the information required to access a user's account such as

name, email address, organization, name of the resource on which the account was discovered, and so on, plus any additional attributes the organization wants stored in the index entry (e.g., if the organization wants to be able to search the index entry by user name, they may also wish to store attributes such as "firstname," "lastname," or "fullname"). Other resource attributes mapped to a virtual attribute by the schema map can be held in RAM (or other volatile memory) rather than saved in identity index 310.

[0059] Software program 240, at step 524, can discover or locate additional resource accounts for the users identities discovered at step 522. Using the resource definitions provided at step 516, software program 240 can connect with each resource and generate possible matching resource accounts. Again, for a user discovered in step 522 (e.g., user Jane Doe), software program 240 could interrogate email server 215 and find possible matching accounts (e.g., jdoe@organization.com, doej@organization, jdoe2@orgnanization.com) by generating a list of likely account names using any account name rule defined for the resource. Additionally, software program 240 can apply the correlation rules defined by administrator 260 to further select accounts. If, for example, on email server 215, each email account also had the employees name or identification number as an attribute, software program 240 could compare the employee ID number for an account matching Jane Doe found on another resource (e.g., the source resource or other earlier interrogated resource) based on the schema map. Alternatively, software program 240 can locate user resource accounts by correlation rules alone. For example, software program 240 can search email system 215 for accounts based on an "employee identification" attribute, or another resource-specific account attribute, without generating possible matches using account name rules. When a matching account is found, the resource name on which the account was found and the attributes for the matching account, such as account name, organization and any other attributes the organization wants to be stored, can be stored under the user's virtual identity in identity index 310.

[0060] In the above discussion, every resource identified by administrator 260 is interrogated for accounts for each user. However, administrator 260 can assign user roles that can be used to limit the resources scanned for a particular user. Each role implies a set of resources to which the user should have access. Embodiments of the discovery process can include searching for accounts only on the resources to which a user has access.

[0061] A role can be assigned to a user in several ways. This can be done by mapping a resource attribute to a "virtual" role attribute. For example, if the source resource contained a "job title" attribute, administrator 260 could map "job title" to the virtual "role" attribute. When a user's account is first imported from the source resource by software 240, the user's role can be assigned based on the job title. Alternatively, administrator 260 can specify a role as part of the import task. In such a case, the specified role is assigned to each user created by a particular import, regardless of any extracted account attributes. Finally, an administrator can specify a role when creating new user (e.g., a user not previously defined by a resource) or when editing an existing user.

[0062] To summarize steps 500-524. Administrator 260 can identify resources on system 200 and provide sufficient information to software program 240 so that software program 240 can access the resources. Administrator 260 can also define a schema map for each resource (or class of resources if each instance of a resource type is identical). The schema map for a resource maps one or more attributes from the information objects defined on the resource to one or more virtual attributes for a user. In addition, administrator 260 can define an identity template which can generate a resource-specific name (e.g., an account name) given a virtual identity and its attributes. At steps 516 and 518, administrator 260 can define resource name and correlation rules to aid in associating resource information objects with particular users. At step 520, administrator 260 can define one or more source resources from which one or more users can be derived based on the information objects contained on the source resource(s). Software program 240, at step 522, can interrogate the source resources and create a virtual identity for each user discovered from the source resource(s). For each user, software program 240 can save information object attributes, as chosen by administrator 260, from the source resource under the user's virtual identity. At step 524, software program 240 can interrogate the other resources defined by administrator 260 to find accounts that correspond to each user. Software program 240 can locate matching accounts based on account name rules and/or correlation rules and save attributes from the accounts under the user's virtual identity. In this manner, software program 240 can associate information objects (e.g., accounts) with virtual identities (e.g., users).

[0063] In addition to administrator 260 defining and identifying resources, the process of defining and identifying resources, at step 540, can be delegated to other administrators (e.g.,

departmental administrators or junior administrators). In the context of FIGURE 2, for example, first departmental administrator 262 and second departmental administrator 264 can identify and define second Unix system 216 and third Unix system 218, respectively. Delegation allows resources to be identified and defined that may not be known to administrator 260, increasing the robustness and efficiency of embodiments of the embodiments of the present invention.

[0064] As a further step in delegation, the end-user may also be given authority to identify resources on which he or she has an account (step 542). The end user can select the resource on which the account resides, and then supply the account name. This allows users to locate stray accounts. For example, if Jane Doe works under first departmental administrator 262, but also occasionally works with employees under second departmental administrator 264, she may have accounts on both second Unix system 216 and third Unix system 218. While first departmental administrator 262 may know of her account on second Unix system 218, neither first departmental administrator 262 or second departmental administrator 264 may know of her account on third Unix system 218. This might occur because first departmental administrator 262 has no knowledge of third Unix system 218 and second departmental administrator 264 has no knowledge of Jane Doe. In such a case, Jane Doe may wish to associate her account on third Unix system 218 with herself at her virtual identity and can be given authority to do so (e.g., discovery of accounts can be further delegated to Jane Doe). In one embodiment of the present invention, however, Jane Doe may have to prove ownership of the account on third Unix system 218 through password authentication, for example, before the account will be associated with her in her virtual identity. This, combined with the functional restriction that an end-user is allowed to update only that end-user's virtual identity, permits secure delegation to an end-user.

[0065] Embodiments of the present invention have been described primarily in terms of user account management, however, it should be understood that the present invention is equally applicable to discovering any set of resource-specific information objects that together comprise a virtual identity. Embodiments of the present invention have been further described in terms of using and/or creating an identity index in which information objects are not replicated at a centralized database. However, it should also be understood that embodiments of the present invention can be used to extract and correlate accounts and other information objects to load into a centralized database or to populate a directory or meta-directory.

[0066] Embodiments of the present invention can comprise the steps of (i) defining at least one source resource, such as an email server, containing a set of information objects (e.g., accounts), wherein the set of information objects defines a set of users; (ii) discovering said set of users from said source resource; (iii) defining an additional resource containing a second set of information objects, wherein each information object from said second set of information objects corresponds to a user from said set of users; (iv)automatically discovering said second set of information objects from said additional resource; and (v) associating each information object from said second set of information objects with the corresponding user from said set of users.

[0067] Another embodiment of the present invention comprises (i) defining a first resource containing information objects defining a set of users; (ii) discovering said information objects based on said first resource definition; (iii) associating each of said information objects with a user from said set of users and with said first resource.

[0068] Yet another embodiment of the method of the present invention includes a method comprises the steps of (i) receiving a first resource definition from a first administrator, wherein said first resource contains a first set of information objects defining at least one user from a set of users; (ii) receiving a second resource definition from a second administrator, wherein said second resource contains a second set of information objects defining at least one user from said set of users; (iii) discovering said first set of information objects from said first resource; (iv) associating each information object from said first set of information objects with at least one user from said set of users and with said first resource; (v) discovering said second set of information objects from said second resource; and (vi) associating each information object from said second set of information objects with at least one user from said set of users and with said second resource.

[0069] Yet another embodiment of the present invention comprises the steps of (i) defining a plurality of resources, wherein each of the plurality of resources contains a set of information objects defining at least one user from a set of users; (ii) discovering said set of information objects from each of the plurality of resources; and (iii) associating each information object from said set of information objects with a user from said set of users and with the resource from which the corresponding information object was discovered.

[0070] Although the present invention has been described in detail herein with reference to the illustrative embodiments, it should be understood that the description is by of example only and is not to be construed in a limiting sense. It is to be further understood, therefore, that numerous changes in the details of the embodiments of this invention an additional embodiments of this invention will be apparent to, and may be made by, persons of ordinary skill in the art having reference to this description. It is contemplated that all such changes and additional embodiments are with the scope of this invention as claimed below.